Corso di Architettura degli Elaboratori e Laboratorio (M-Z)

Rappresentazione binaria di dati complessi

Nino Cauli

UNIVERSITÀ
degli STUDI
di CATANIA

Dipartimento di Matematica e Informatica

Numeri frazionari

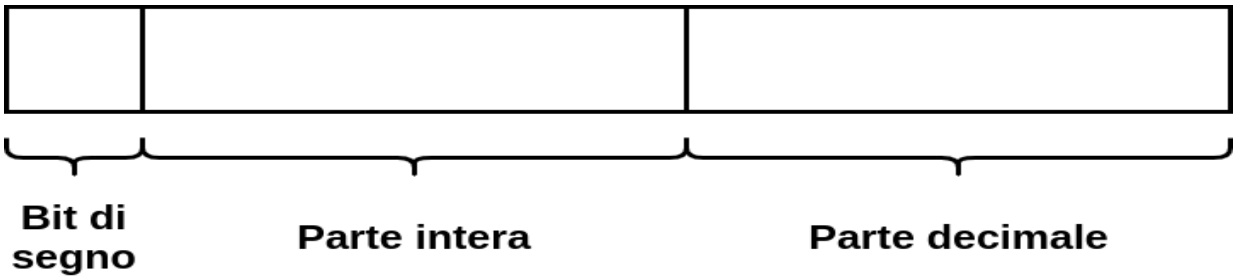
Come rappresentare i **numeri reali** in binario?

Idea semplice:

- Un **bit di segno**
- Una **porzione di bit fissa** per la **parte intera**
- Una **porzione di bit fissa** per la **parte decimale**

Chiamiamo questa rappresentazione **A VIRGOLA FISSA (FIXED POINT)**

Numeri a virgola fissa (fixed point)



Solo bit di segno e parte intera:

- Valori rappresentabili: da -2^{n-1} a $2^{n-1} - 1$
- Risoluzione: **1**

Solo bit di segno e parte decimale:

- Valori rappresentabili: da **-1** a $1 - 2^{-(n-1)}$
- Risoluzione: $2^{-(n-1)}$

Intervallo non sufficiente per calcoli scientifici.

Numeri virgola mobile (floating point)

Per aumentare intervallo e risoluzione di valori rappresentabili si potrebbe spostare la posizione della virgola dinamicamente (**VIRGOLA MOBILE**)

Notazione scientifica decimale, **FORMA NORMALE**:

$$6{,}0247 \times 10^{2} = 602{,}47 \qquad 3{,}7291 \times 10^{-2} = 0{,}037291$$

In generale vale per **ogni base**:

$$1{,}0011 \times 2^{2} = 100{,}11 \qquad 4{,}2131 \times 5^{-2} = 0{,}042131$$

Numeri frazionari

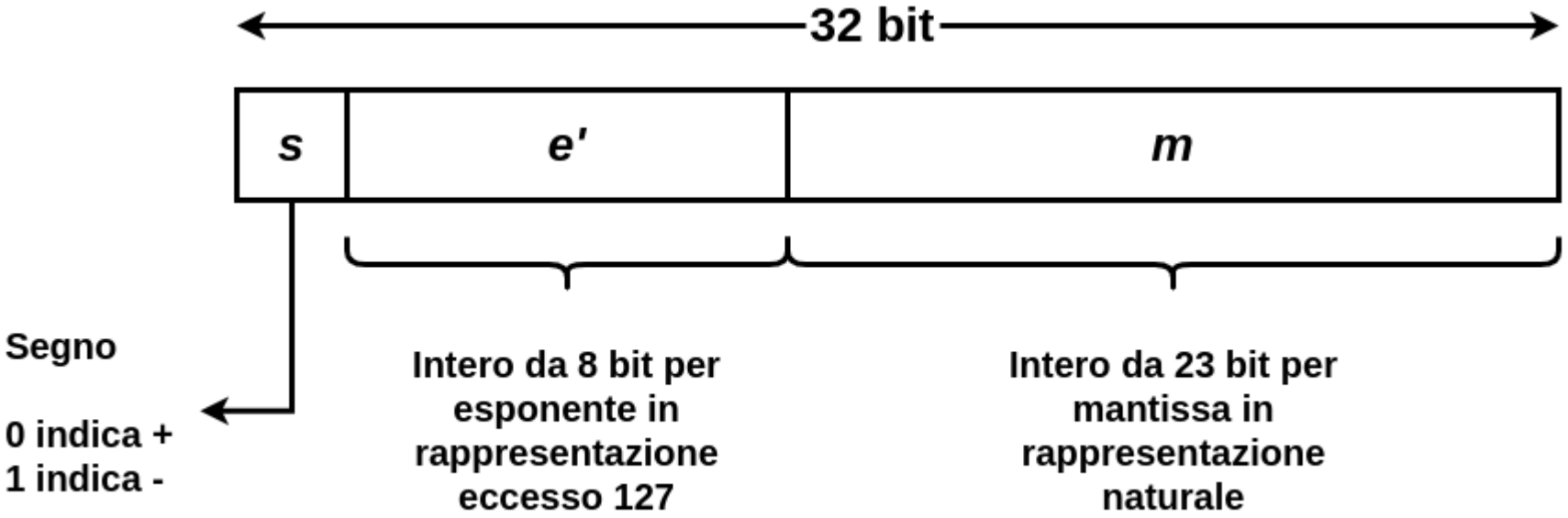
Un numero binario in virgola mobile può quindi essere rappresentato:

- Un **SEGNO** *s* per il numero
- La **MANTISSA** *m* (bit significativi escluso il bit più significativo)
- Un **ESPONENTE** *e* con segno in base 2

$$\text{Valore rappresentato} = \pm\, \mathbf{1, \boldsymbol{m}} \times \mathbf{2^{\boldsymbol{e}}}$$

Formato precisione singola (32 bit)

Standard **IEEE 754** numeri **32 bit**



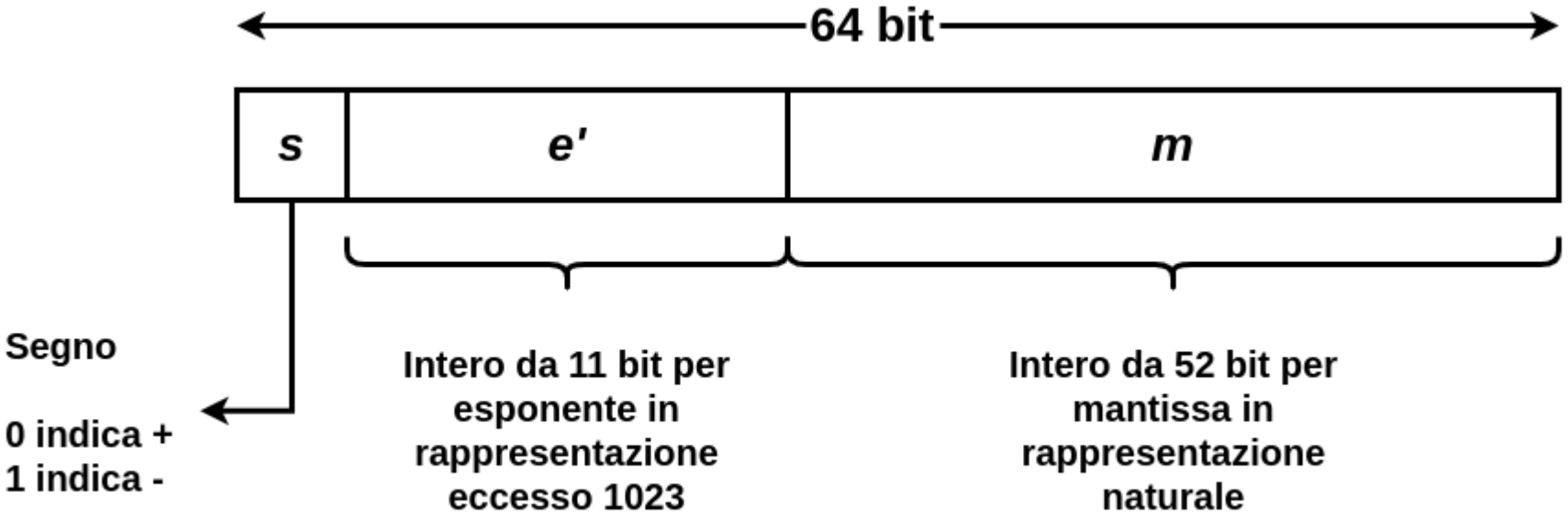
$0 \leq e' \leq 255$ $\quad e = e' - 127$

Valori speciali: $e' = 0$, $e' = 255$ Intervallo esponente: $-126 \leq e \leq 127$

Fattore di scala nell'intervallo: $[2^{-126}, 2^{127}]$

Formato precisione doppia (64 bit)

Standard **IEEE 754** numeri **64 bit**



$0 \leq e' \leq 2047$ $e = e' - 1023$

Valori speciali: $e' = 0$, $e' = 2047$ Intervallo esponente: $-1022 \leq e \leq 1023$

Fattore di scala nell'intervallo: $[2^{-1022}, 2^{1023}]$

Valori speciali

Alcuni valori dell'esponente sono speciali:

- **$e' = 0, m = 0$** rappresenta lo **0 esatto**
- **$e' = 255(2047), m = 0$** rappresenta l'infinito ∞
- **$e' = 0, m \neq 0$** rappresenta la **forma non normale**: **$\pm 0,m \times 2^{-126(-1022)}$**
- **$e' = 255(2047), m \neq 0$** rappresenta **Not a Number NaN**

Rappresentazione dei caratteri

UNIVERSITÀ degli STUDI di CATANIA

Come rappresentare caratteri tramite una sequenza di *n* bit?

- Associamo un carattere ad ogni possibile valore binario rappresentabile

Quanti caratteri siamo in grado di rappresentare con *n* bit?

- Una sequenza di ***n* bit** può rappresentare **2^n permutazioni di 0 e 1**
- Si può rappresentare un alfabeto di **2^n simboli**

Vediamo gli standard più usati

Codice ASCII

- **Codice ASCII** (American Standard Code for Information Interchange)
- Rappresenta lettere, cifre decimali, punteggiatura e caratteri speciali
- Definito su **7 bit** → alfabeto di **2^7 = 128 elementi**
- Lettere e numeri con codici in ordine crescente

	Bit 654							
Bit 3210	000	001	010	011	100	101	110	111
0000	NUL	DLE	SPACE	0	@	P	‘	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	”	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	’	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	/	l	|
1101	CR	GS	–	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

Standard per alfabeti con più simboli

UNIVERSITÀ degli STUDI di CATANIA

Necessità di codici più ricchi per gestire le diverse lingue con caratteri speciali, accenti, etc.

Standard internazionali:

- Famiglia **ISO 8859-x**: estendono il codice ASCII usando 8 bit (doppio dei simboli)
- **ISO/IEC 10646 (UCS)**: rappresentazione universale di caratteri che estende su più byte la ISO 8859
- Standard di codifica basati su UCS: come ad esempio **UNICODE** e **UTF-8**